

LAPORAN TUGAS KECIL 1  
IF2211 STRATEGI ALGORITMA  
SEMESTER II TAHUN 2021/2022

**Penyelesaian Word Search Puzzle dengan  
Algoritma Brute Force**

Raka Wirabuana Ninagan || 13520134 || K-02

Tugas ini merupakan tugas untuk mengimplementasikan algoritma *brute force*, berupa pembuatan *Word Search Puzzle*.

**1. Algoritma Brute Force**

Misalkan program memiliki matriks yang berisi huruf-huruf. Beberapa urutan huruf dengan berbagai variasi arahnya merupakan kata yang diinginkan, sehingga kata-kata tersebut disimpan di *array of strings*. Pendekatan yang dilakukan oleh saya adalah memilih arah pengecekan sebagai dasar pergerakan pencocokan strings, yang saya implementasikan sebagai 8 arah (kanan, kiri, atas, bawah, kanan bawah, kanan atas, kiri bawah, kiri atas). Setiap kata yang dicari akan dicek sesuai giliran arah, contohnya, ada 5 kata yang dicari, satu arah akan cek kelima kata tersebut, lalu ganti arah, cek kelima kata tersebut, ulangi sampai arah yang paling terakhir.

Pada pergerakan arah horizontal, setiap baris akan dicek apakah ada dari kata-kata yang dicari terbentuk di baris tersebut. Ketika tidak ada, maka akan lanjut ke baris selanjutnya sampai paling bawah.

Pada pergerakan arah vertikal, secara algoritma mirip dengan yang horizontal, tetapi acuan pengecekannya berdasarkan kolom. Tiap kolom dicek keberadaan kata-kata yang dicari sampai kolom terakhir.

Pada pergerakan arah diagonal, pendekatan yang dilakukan adalah membagi pergerakan pengecekan menjadi 2, yaitu dari indeks paling awal/akhir ke kanan/kiri dilanjutkan dengan pergerakan ke atas/bawah (untuk bentuk persegi panjang mendatar), dan dari indeks paling awal/akhir ke atas/bawah dilanjutkan dengan pergerakan ke kanan/kiri (untuk bentuk persegi panjang tegak). Contoh, misalkan ada matriks dengan ukuran persegi panjang 5 x 4 (indeks baris dari 0-4 dan kolom dari 0-3). Maka pergerakan pengecekan per diagonalnya adalah (0,0) ke (0,1) ke (0,2) ke (0,3), lalu geser baris ke indeks 1 sehingga lanjutannya adalah (1,0) ke (2,0) ke (3,0). Setiap indeks diatas adalah acuan untuk melakukan pengecekan diagonal, sehingga tiap indeks baris atau kolom dijumlahkan 1.

Tiap keberhasilan pencocokan akan ditulis ke layar beserta posisi melalui visualisas matriks. Ketika kata yang dicari berhasil ditemukan, maka kata tersebut akan dihilangkan dari array kata agar tidak dapat dicek kembali (mengingat bahwa 1 kata cukup hanya mencari 1 kali ditemukan saja) dengan tujuan menghemat waktu.

## 2. Source Program

### Main.java

```
import java.util.Scanner;
import java.io.FileNotFoundException;

public class Main {
    public static void main(String[] args) throws FileNotFoundException{

        System.out.println("| WORD SEARCH PUZZLE |\n");
        System.out.println("Masukkan nama file .txt yang diinginkan. (contoh
ketikan: large1.txt)");
        System.out.println("Mohon pastikan tidak ada spasi di tempat yang tidak
sesuai aturan masukan.\n");

        // Membaca nama file (beserta extensionnya)
        // Contoh : Large1.txt
        Scanner input = new Scanner(System.in);
        System.out.print("Masukkan nama file : ");
        String file = input.nextLine();

        input.close();
        System.out.println();

        // Mendeklarasikan matriks dan array sekaligus memasukkan isinya
        // Detail file processing ada di file FileProcess.java

        char[][] puzzlwordmatrix = FileProcess.matrixProcessing(file);
        String[] targetwordlist = FileProcess.arrayProcessing(file);

        System.out.println("Masukan Puzzle :");
        displayMat(puzzlwordmatrix, "-");

        System.out.println("Daftar kata yang dicari :");

        for (int s = 0; s < targetwordlist.length; s++){
            System.out.println("- " + targetwordlist[s]);
        }
        for (int s = 0; s < puzzlwordmatrix[0].length; s++){
            System.out.print("---");
```

```

    }
    System.out.println();
    System.out.println("HASIL PENCARIAN :");

    // Mendeklarasikan matrix display dan inisialisasi dengan simbol -
    char[][] foundedMat = new
char[puzzlewordmatrix.length][puzzlewordmatrix[0].length];
    initializeMat(foundedMat);

    // Mengaktifkan Penghitung Waktu
    long start = System.nanoTime();
    targetwordlist = horizontalRight(puzzlewordmatrix, foundedMat,
targetwordlist);
    System.out.println();

    // Horizontal Kiri
    targetwordlist = horizontalLeft(puzzlewordmatrix, foundedMat,
targetwordlist);
    System.out.println();

    // Vertikal Bawah
    targetwordlist = verticalDown(puzzlewordmatrix, foundedMat,
targetwordlist);
    System.out.println();

    // Vertikal Atas
    targetwordlist = verticalUp(puzzlewordmatrix, foundedMat,
targetwordlist);
    System.out.println();

    // Kanan Bawah
    targetwordlist = southEast(puzzlewordmatrix, foundedMat, targetwordlist);
    System.out.println();

    // Kanan Atas
    targetwordlist = northEast(puzzlewordmatrix, foundedMat, targetwordlist);
    System.out.println();

    // Kiri Bawah
    targetwordlist = southWest(puzzlewordmatrix, foundedMat, targetwordlist);
    System.out.println();

    // Kiri Atas
    targetwordlist = northWest(puzzlewordmatrix, foundedMat, targetwordlist);
    System.out.println();

```

```

        // Mencatat durasi kerja pencocokan dan penampilan matriks ke layar
        long elapsedTime = System.nanoTime() - start;
        double seconds = (double)elapsedTime / 1_000_000_000.0;
        System.out.print("Matching time : ");
        System.out.print(seconds);
        System.out.print(" s.");
    }

```

```

    // Menampilkan matrix
    public static void displayMat(char[][] matrix, String word){
        System.out.println("> WORD : " + word);
        for (int i = 0; i < matrix.length; i++){
            System.out.print(" |");
            for (int j = 0; j < matrix[0].length; j++){
                System.out.print(" " + matrix[i][j] + " ");
            }
            System.out.print("|");
            System.out.println();
        }
        System.out.println();
    }

```

```

    // Menginisialisasi matrix untuk hasil pencocokan kata
    // Mengembalikan matrix display yang terisi menjadi seperti semula
    public static void initializeMat(char[][] matrix){
        for (int i = 0; i < matrix.length; i++){
            for (int j = 0; j < matrix[0].length; j++){
                if (matrix[i][j] != '-'){
                    matrix[i][j] = '-';
                }
            }
        }
    }

```

```

    // Delete elemen array setelah ditemukan
    public static String[] deleteWord(String[] wordlist, String word){
        String[] newWordlist = new String[wordlist.length];
        int i = 0, j = 0;
        while (i < newWordlist.length){
            if (wordlist[j] != word){
                newWordlist[i] = wordlist[j];
                j++;
            }
            else{

```

```

        newWordlist[i] = null;
        j += 1;
    }
    i++;
}
return newWordlist;
}

// Horizontal Kanan
public static String[] horizontalRight(char[][] puzzlemat, char[][]
resultmat, String[] targetarr){
    System.out.println(">> Horizontal Right direction");
    boolean filled = false;
    int jumlahpengecekan = 0;
    for (int g = 0; g < targetarr.length; g++){
        if (targetarr[g] != null){
            String targetword = targetarr[g];
            int twlength = targetword.length();
            boolean found = false;

            int h = 0;
            while ((h < puzzlemat.length) && (!found)){
                int i = 0;
                while (i <= puzzlemat[0].length - twlength && (!found)){
                    int j = 0;
                    while ((j < twlength) && (puzzlemat[h][i+j] ==
targetword.charAt(j)) && (!found)){
                        filled = true;
                        resultmat[h][i+j] = puzzlemat[h][i+j];
                        j += 1;
                    }
                    if (j == twlength){
                        displayMat(resultmat, targetword);
                        initializeMat(resultmat);
                        targetarr[g] = null;
                        found = true;
                    }
                }
                if (filled){
                    initializeMat(resultmat);
                    filled = false;
                }
                jumlahpengecekan = jumlahpengecekan + j + 1;
                i++;
            }
            h++;
        }
    }
}

```

```

    }
}
}
System.out.println("Jumlah huruf yang dicek : " + jumlahpengecekan);
return targetarr;
}

// Horizontal Kiri
public static String[] horizontalLeft(char[][] puzzlemat, char[][] resultmat,
String[] targetarr){
    System.out.println(">> Horizontal Left direction");
    boolean filled = false;
    int jumlahpengecekan = 0;
    for (int g = 0; g < targetarr.length; g++){
        if (targetarr[g] != null){
            String targetword = targetarr[g];
            int twlength = targetword.length();
            boolean found = false;

            int h = 0;
            while ((h < puzzlemat.length) && (!found)){
                int i = puzzlemat[0].length - 1;
                while (i >= twlength - 1){
                    int j = 0;
                    while ((j < twlength) && (puzzlemat[h][i-j] ==
targetword.charAt(j)) && (!found)){
                        filled = true;
                        resultmat[h][i-j] = puzzlemat[h][i-j];
                        j += 1;
                    }
                    if (j == twlength){
                        displayMat(resultmat, targetword);
                        initializeMat(resultmat);
                        targetarr[g] = null;
                        found = true;
                    }
                }
                if (filled){
                    initializeMat(resultmat);
                    filled = false;
                }
                jumlahpengecekan = jumlahpengecekan + j + 1;
                i -= 1;
            }
            h++;
        }
    }
}

```

```

    }
}
System.out.println("Jumlah huruf yang dicek : " + jumlahpengecekan);
return targetarr;
}

// Vertikal Bawah
public static String[] verticalDown(char[][] puzzlemat, char[][] resultmat,
String[] targetarr){
    System.out.println(">> Vertical Down direction");
    boolean filled = false;
    int jumlahpengecekan = 0;
    for (int g = 0; g < targetarr.length; g++){
        if (targetarr[g] != null){
            String targetword = targetarr[g];
            int twlength = targetword.length();
            boolean found = false;

            int h = 0;
            while ((h < puzzlemat[0].length) && (!found)){
                int i = 0;
                while (i <= puzzlemat.length - twlength){
                    int j = 0;
                    while ((j < twlength) && (puzzlemat[i+j][h] ==
targetword.charAt(j)) && (!found)){
                        filled = true;
                        resultmat[i+j][h] = puzzlemat[i+j][h];
                        j += 1;
                    }
                    if (j == twlength){
                        displayMat(resultmat, targetword);
                        initializeMat(resultmat);
                        targetarr[g] = null;
                        found = true;
                    }
                    if (filled){
                        initializeMat(resultmat);
                        filled = false;
                    }
                    jumlahpengecekan = jumlahpengecekan + j + 1;
                    i++;
                }
                h++;
            }
        }
    }
}

```

```

    }
    System.out.println("Jumlah huruf yang dicek : " + jumlahpengecekan);
    return targetarr;
}

// Vertikal Atas
public static String[] verticalUp(char[][] puzzlemat, char[][] resultmat,
String[] targetarr){
    System.out.println(">> Vertical Up direction");
    boolean filled = false;
    int jumlahpengecekan = 0;
    for (int g = 0; g < targetarr.length; g++){
        if (targetarr[g] != null){
            String targetword = targetarr[g];
            int twlength = targetword.length();
            boolean found = false;

            int h = 0;
            while ((h < puzzlemat[0].length) && (!found)){
                int i = puzzlemat.length - 1;
                while (i >= twlength - 1){
                    int j = 0;
                    while ((j < twlength) && (puzzlemat[i-j][h] ==
targetword.charAt(j)) && (!found)){
                        filled = true;
                        resultmat[i-j][h] = puzzlemat[i-j][h];
                        j += 1;
                    }
                    if (j == twlength){
                        displayMat(resultmat, targetword);
                        initializeMat(resultmat);
                        targetarr[g] = null;
                        found = true;
                    }
                    if (filled){
                        initializeMat(resultmat);
                        filled = false;
                    }
                    jumlahpengecekan = jumlahpengecekan + j + 1;
                    i--;
                }
                h++;
            }
        }
    }
}

```



```

        System.out.println("Jumlah huruf yang dicek : " + jumlahpengecekan);
        return targetarr;
    }

    // Arah tenggara
    public static String[] southEast(char[][] puzzlemat, char[][] resultmat,
String[] targetarr){
        System.out.println(">> Southeast direction");
        boolean filled = false;
        int jumlahpengecekan = 0;
        for (int g = 0; g < targetarr.length; g++){
            if (targetarr[g] != null){
                String targetword = targetarr[g];
                int twlength = targetword.length();
                boolean found = false;

                if (puzzlemat.length > puzzlemat[0].length){
                    int f = 0;
                    while (f < puzzlemat[0].length && (!found)){
                        int h = 0, i = f;
                        while (h <= puzzlemat[0].length - twlength && i <=
puzzlemat.length - twlength && (!found)){
                            int j = 0;
                            while ((j < twlength) && (puzzlemat[h+j][i+j] ==
targetword.charAt(j)) && (!found)){
                                filled = true;
                                resultmat[h+j][i+j] = puzzlemat[h+j][i+j];
                                j += 1;
                            }
                            if (j == twlength){
                                displayMat(resultmat, targetword);
                                initializeMat(resultmat);
                                targetarr[g] = null;
                                found = true;
                            }
                        }
                        if (filled){
                            initializeMat(resultmat);
                            filled = false;
                        }
                        jumlahpengecekan = jumlahpengecekan + j + 1;
                        h++;
                        i++;
                    }
                    f++;
                }
            }
        }
    }

```



```

        initializeMat(resultmat);
        targetarr[g] = null;
        found = true;
    }
    if (filled){
        initializeMat(resultmat);
        filled = false;
    }
    jumlahpengecekan = jumlahpengecekan + j + 1;
    h++;
    i++;
}
f++;
}
f = 1;
while (f < puzzlemat.length && (!found)){
    int h = f, i = 0;
    while (h <= puzzlemat.length - twlength && i <=
puzzlemat[0].length - twlength && (!found)){
        int j = 0;

        while ((j < twlength) && (puzzlemat[h+j][i+j] ==
targetword.charAt(j))){
            filled = true;
            resultmat[h+j][i+j] = puzzlemat[h+j][i+j];
            j += 1;
        }
        if (j == twlength){
            displayMat(resultmat, targetword);
            initializeMat(resultmat);
            targetarr[g] = null;
            found = true;
        }
        if (filled){
            initializeMat(resultmat);
            filled = false;
        }
        jumlahpengecekan = jumlahpengecekan + j + 1;
        h++;
        i++;
    }
    f++;
}
}
}
}

```

```

    }
    System.out.println("Jumlah huruf yang dicek : " + jumlahpengecekan);
    return targetarr;
}

// Arah timur laut
public static String[] northEast(char[][] puzzlemat, char[][] resultmat,
String[] targetarr){
    System.out.println(">> Northeast direction");
    boolean filled = false;
    int jumlahpengecekan = 0;
    for (int g = 0; g < targetarr.length; g++){
        if (targetarr[g] != null){
            String targetword = targetarr[g];
            int twlength = targetword.length();
            boolean found = false;

            if (puzzlemat.length > puzzlemat[0].length){
                int f = puzzlemat.length - 1;
                while (f >= twlength - 1 && (!found)){
                    int h = f, i = 0;
                    while (h >= twlength - 1 && i <= puzzlemat[0].length -
twlength && (!found)){
                        int j = 0;
                        while ((j < twlength) && (puzzlemat[h-j][i+j] ==
targetword.charAt(j)) && (!found)){
                            filled = true;
                            resultmat[h-j][i+j] = puzzlemat[h-j][i+j];
                            j += 1;
                        }
                        if (j == twlength){
                            displayMat(resultmat, targetword);
                            initializeMat(resultmat);
                            targetarr[g] = null;
                            found = true;
                        }
                    }
                    if (filled){
                        initializeMat(resultmat);
                        filled = false;
                    }
                    jumlahpengecekan = jumlahpengecekan + j + 1;
                    h--;
                    i++;
                }
                f--;
            }
        }
    }
}

```

```

    }
    f = 1;
    while (f < puzzlemat[0].length && (!found)){
        int h = puzzlemat.length - 1, i = f;
        while (h >= twlength - 1 && i <= puzzlemat[0].length -
twlength && (!found)){
            int j = 0;

            while ((j < twlength) && (puzzlemat[h-j][i+j] ==
targetword.charAt(j))){
                filled = true;
                resultmat[h-j][i+j] = puzzlemat[h-j][i+j];
                j += 1;
            }
            if (j == twlength){
                displayMat(resultmat, targetword);
                initializeMat(resultmat);
                targetarr[g] = null;
                found = true;
            }
            if (filled){
                initializeMat(resultmat);
                filled = false;
            }
            jumlahpengecekan = jumlahpengecekan + j + 1;
            h--;
            i++;
        }
        f++;
    }
}
else{
    int f = 0;
    while (f < puzzlemat[0].length && (!found)){
        int h = puzzlemat.length - 1, i = f;
        while (h >= twlength - 1 && i <= puzzlemat[0].length -
twlength && (!found)){
            int j = 0;
            while ((j < twlength) && (puzzlemat[h-j][i+j] ==
targetword.charAt(j)) && (!found)){
                filled = true;
                resultmat[h-j][i+j] = puzzlemat[h-j][i+j];
                j += 1;
            }
            if (j == twlength){

```

```

        displayMat(resultmat, targetword);
        initializeMat(resultmat);
        targetarr[g] = null;
        found = true;
    }
    if (filled){
        initializeMat(resultmat);
        filled = false;
    }
    jumlahpengecekan = jumlahpengecekan + j + 1;
    h--;
    i++;
}
f++;
}
f = puzzlemat.length - 2;
while (f >= twlength - 1 && (!found)){
    int h = f, i = 0;
    while (h >= twlength - 1 && i <= puzzlemat[0].length -
twlength && (!found)){
        int j = 0;

        while ((j < twlength) && (puzzlemat[h-j][i+j] ==
targetword.charAt(j))){
            filled = true;
            resultmat[h-j][i+j] = puzzlemat[h-j][i+j];
            j += 1;
        }
        if (j == twlength){
            displayMat(resultmat, targetword);
            initializeMat(resultmat);
            targetarr[g] = null;
            found = true;
        }
        if (filled){
            initializeMat(resultmat);
            filled = false;
        }
        jumlahpengecekan = jumlahpengecekan + j + 1;
        h--;
        i++;
    }
    f--;
}
}
}

```

```

    }
}
System.out.println("Jumlah huruf yang dicek : " + jumlahpengecekan);
return targetarr;
}

// Arah barat daya
public static String[] southWest(char[][] puzzlemat, char[][] resultmat,
String[] targetarr){
    System.out.println(">> Southwest direction");
    boolean filled = false;
    int jumlahpengecekan = 0;
    for (int g = 0; g < targetarr.length; g++){
        if (targetarr[g] != null){
            String targetword = targetarr[g];
            int twlength = targetword.length();
            boolean found = false;

            if (puzzlemat.length > puzzlemat[0].length){
                int f = 0;
                while (f < puzzlemat.length && (!found)){
                    int h = f, i = puzzlemat[0].length - 1;
                    while (h <= puzzlemat.length - twlength && i >= twlength
- 1 && (!found)){
                        int j = 0;
                        while ((j < twlength) && (puzzlemat[h+j][i-j] ==
targetword.charAt(j)) && (!found)){
                            filled = true;
                            resultmat[h+j][i-j] = puzzlemat[h+j][i-j];
                            j += 1;
                        }
                        if (j == twlength){
                            displayMat(resultmat, targetword);
                            initializeMat(resultmat);
                            targetarr[g] = null;
                            found = true;
                        }
                    }
                    if (filled){
                        initializeMat(resultmat);
                        filled = false;
                    }
                    jumlahpengecekan = jumlahpengecekan + j + 1;
                    h++;
                    i--;
                }
            }
        }
    }
}

```

```

        f++;
    }
    f = puzzlemat[0].length - 2;
    while (f >= twlength - 1 && (!found)){
        int h = 0, i = f;
        while (h <= puzzlemat.length - twlength && i >= twlength
- 1 && (!found)){
            int j = 0;

            while ((j < twlength) && (puzzlemat[h+j][i-j] ==
targetword.charAt(j))){
                filled = true;
                resultmat[h+j][i-j] = puzzlemat[h+j][i-j];
                j += 1;
            }
            if (j == twlength){
                displayMat(resultmat, targetword);
                initializeMat(resultmat);
                targetarr[g] = null;
                found = true;
            }
            if (filled){
                initializeMat(resultmat);
                filled = false;
            }
            jumlahpengecekan = jumlahpengecekan + j + 1;
            h++;
            i--;
        }
        f--;
    }
}
else{
    int f = puzzlemat[0].length - 1;
    while (f >= twlength - 1 && (!found)){
        int h = 0, i = f;
        while (h <= puzzlemat.length - twlength && i >= twlength
- 1 && (!found)){
            int j = 0;
            while ((j < twlength) && (puzzlemat[h+j][i-j] ==
targetword.charAt(j)) && (!found)){
                filled = true;
                resultmat[h+j][i-j] = puzzlemat[h+j][i-j];
                j += 1;
            }
        }
    }
}

```



```

        if (j == twlength){
            displayMat(resultmat, targetword);
            initializeMat(resultmat);
            targetarr[g] = null;
            found = true;
        }
        if (filled){
            initializeMat(resultmat);
            filled = false;
        }
        jumlahpengecekan = jumlahpengecekan + j + 1;
        h++;
        i--;
    }
    f--;
}
f = 1;
while (f < puzzlemat.length && (!found)){
    int h = f, i = puzzlemat[0].length - 1;
    while (h <= puzzlemat.length - twlength && i >= twlength
- 1 && (!found)){
        int j = 0;

        while ((j < twlength) && (puzzlemat[h+j][i-j] ==
targetword.charAt(j))){
            filled = true;
            resultmat[h+j][i-j] = puzzlemat[h+j][i-j];
            j += 1;
        }
        if (j == twlength){
            displayMat(resultmat, targetword);
            initializeMat(resultmat);
            targetarr[g] = null;
            found = true;
        }
        if (filled){
            initializeMat(resultmat);
            filled = false;
        }
        jumlahpengecekan = jumlahpengecekan + j + 1;
        h++;
        i--;
    }
    f++;
}

```

```

    }
}
}
System.out.println("Jumlah huruf yang dicek : " + jumlahpengecekan);
return targetarr;
}

// Arah Barat Laut
public static String[] northWest(char[][] puzzlemat, char[][] resultmat,
String[] targetarr){
    System.out.println(">> Northwest direction");
    boolean filled = false;
    int jumlahpengecekan = 0;
    for (int g = 0; g < targetarr.length; g++){
        if (targetarr[g] != null){
            String targetword = targetarr[g];
            int twlength = targetword.length();
            boolean found = false;

            if (puzzlemat.length > puzzlemat[0].length){
                int f = puzzlemat.length - 1;
                while (f >= twlength - 1 && (!found)){
                    int h = f, i = puzzlemat[0].length - 1;
                    while (h >= twlength - 1 && i >= twlength - 1 &&
(!found)){
                        int j = 0;
                        while ((j < twlength) && (puzzlemat[h-j][i-j] ==
targetword.charAt(j)) && (!found)){
                            filled = true;
                            resultmat[h-j][i-j] = puzzlemat[h-j][i-j];
                            j += 1;
                        }
                        if (j == twlength){
                            displayMat(resultmat, targetword);
                            initializeMat(resultmat);
                            targetarr[g] = null;
                            found = true;
                        }
                        if (filled){
                            initializeMat(resultmat);
                            filled = false;
                        }
                        jumlahpengecekan = jumlahpengecekan + j + 1;
                        h--;
                        i--;
                    }
                }
            }
        }
    }
}

```

```

        }
        f--;
    }
    f = puzzlemat[0].length - 2;
    while (f >= twlength - 1 && (!found)){
        int h = puzzlemat.length - 1, i = f;
        while (h >= twlength - 1 && i >= twlength - 1 &&
(!found)){
            int j = 0;

            while ((j < twlength) && (puzzlemat[h-j][i-j] ==
targetword.charAt(j))){
                filled = true;
                resultmat[h-j][i-j] = puzzlemat[h-j][i-j];
                j += 1;
            }
            if (j == twlength){
                displayMat(resultmat, targetword);
                initializeMat(resultmat);
                targetarr[g] = null;
                found = true;
            }
            if (filled){
                initializeMat(resultmat);
                filled = false;
            }
            jumlahpengecekan = jumlahpengecekan + j + 1;
            h--;
            i--;
        }
        f--;
    }
}
else{
    int f = puzzlemat[0].length - 1;
    while (f >= twlength - 1 && (!found)){
        int h = puzzlemat.length - 1, i = f;
        while (h >= twlength - 1 && i >= twlength - 1 &&
(!found)){
            int j = 0;

            while ((j < twlength) && (puzzlemat[h-j][i-j] ==
targetword.charAt(j))){
                filled = true;
                resultmat[h-j][i-j] = puzzlemat[h-j][i-j];

```

```

        j += 1;
    }
    if (j == twlength){
        displayMat(resultmat, targetword);
        initializeMat(resultmat);
        targetarr[g] = null;
        found = true;
    }
    if (filled){
        initializeMat(resultmat);
        filled = false;
    }
    jumlahpengecekan = jumlahpengecekan + j + 1;
    h--;
    i--;
}
f--;
}
f = puzzlemat.length - 2;
while (f >= twlength - 1 && (!found)){
    int h = f, i = puzzlemat[0].length - 1;
    while (h >= twlength - 1 && i >= twlength - 1 &&
(!found)){
        int j = 0;
        while ((j < twlength) && (puzzlemat[h-j][i-j] ==
targetword.charAt(j)) && (!found)){
            filled = true;
            resultmat[h-j][i-j] = puzzlemat[h-j][i-j];
            j += 1;
        }
        if (j == twlength){
            displayMat(resultmat, targetword);
            initializeMat(resultmat);
            targetarr[g] = null;
            found = true;
        }
        if (filled){
            initializeMat(resultmat);
            filled = false;
        }
        jumlahpengecekan = jumlahpengecekan + j + 1;
        h--;
        i--;
    }
    f--;
}

```

```

    }
    }
}
System.out.println("Jumlah huruf yang dicek : " + jumlahpengecekan);
return targetarr;
}
}

```

### FileProcess.java

```

import java.io.FileReader;
import java.io.IOException;

public class FileProcess {
    public static char[][] matrixProcessing(String nameFile){
        String firstStep = fileInput(nameFile);
        return strToMat(firstStep);
    }

    public static String[] arrayProcessing(String nameFile){
        String firstStep = fileInput(nameFile);
        return strToArr(firstStep);
    }

    public static String fileInput(String strFile)
    {
        String strConv = "";
        String namaFile = "../test/" + strFile;
        try {
            FileReader fRead = new FileReader(namaFile);

            int ch;
            while ((ch = fRead.read()) != -1) {
                strConv += (char)ch;
            }
            fRead.close();
        }
        catch (IOException e) {
            System.out.println("Pembacaan file masukan error.");
        }
        return strConv;
    }

    public static char[][] strToMat(String mat){

```

```

String[] hasilSplit = mat.split("\n");
boolean isRowMax = true;
int i = 0;
while (isRowMax){
    boolean isWhitespace = hasilSplit[i+1].matches("^\\s*$");
    if (isWhitespace){
        isRowMax = false;
        i++;
    }
    else{
        i++;
    }
}
char[][] matrix = new char[i][];
i = 0;

boolean matLooping = true;
while (i < matrix.length && (matLooping)) {
    String[] arr2 = hasilSplit[i].split(" ");
    matrix[i] = new char[arr2.length];

    for (int j = 0; j < matrix[i].length; j++) {
        matrix[i][j] = arr2[j].charAt(0);
    }
    boolean isWhitespace = hasilSplit[i+1].matches("^\\s*$");
    if (isWhitespace){
        matLooping = false;
    }
    i++;
}
return matrix;
}

public static String[] strToArr(String arr){
    String[] hasilSplit = arr.split("\n");
    boolean startWordlist = false;
    int i = 0;
    while (!startWordlist){
        boolean isWhitespace = hasilSplit[i].matches("^\\s*$");
        if (isWhitespace){
            startWordlist = true;
        }
        i++;
    }
    String[] wordList = new String[hasilSplit.length - i];

```

```

        for (int s = 0; s < wordList.length; s++){
            wordList[s] = hasilSplit[i];
            wordList[s] = wordList[s].replaceAll("\\s","");
            i++;
        }
        return wordList;
    }
}

```

### 3. *Screenshot* input dan output

#### 1. Input small1.txt

```

| WORD SEARCH PUZZLE |

Masukkan nama file .txt yang diinginkan. (contoh ketikan: large1.txt)
Mohon pastikan tidak ada spasi di tempat yang tidak sesuai aturan masukan.

Masukkan nama file : small1.txt

```

output :

Masukkan nama file : small1.txt

Masukan Puzzle :

```
>> WORD : -
```

K	V	Y	P	N	P	Q	D	O	H	T	X	Y	I	B	V	F
X	I	K	E	M	H	K	P	Y	D	Z	U	V	E	C	R	O
D	L	W	S	Z	V	W	G	N	O	K	P	R	N	R	A	L
S	R	D	M	N	I	N	O	N	S	T	F	A	L	R	I	E
J	J	H	G	O	I	R	B	I	A	U	M	C	G	O	M	B
J	H	Q	C	N	T	R	U	B	I	R	A	I	N	R	A	K
L	F	K	X	W	Y	A	C	J	H	N	E	F	L	A	M	K
F	I	F	N	W	K	A	H	Q	J	U	N	Y	S	A	M	J
A	Z	Z	W	L	X	P	Y	O	D	A	U	M	B	O	A	N
Z	F	T	U	O	Y	J	S	U	L	Y	K	J	E	H	B	X
L	I	U	R	O	D	A	H	X	W	R	J	Z	I	V	R	C

Daftar kata yang dicari :

- PHANTAMINUM
- ENRYU
- ZAHARD
- MAZINO
- ARIEHON
- KHUNEDUAN
- ADORI
- EURASIA
- BAEKRYUN
- HAYURIN

HASIL PENCARIAN :

```
>> Horizontal Right direction
```

> WORD : KHUNEDUAN

K H U N E D U A N

Jumlah huruf yang dicek : 1787

```
>> Horizontal Left direction
```

```
> WORD : ADORI
```

-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
I	R	O	D	A	-	-	-	-	-	-

```
> WORD : EURASIA
```

[illegible]

> WORD : BAEKRYUN

-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	N	U	Y	R	K	E	A B

Jumlah huruf yang dicek : 1625

```
>> Vertical Down direction
```

```
> WORD : MAZINO
```

[illegible]

```
> WORD : ARIEHON
```

[illegible]

Jumlah huruf yang dicek : 928



```
Jumlah huruf yang dicek : 32
>> Northwest direction
Jumlah huruf yang dicek : 0
Matching time : 0.3611504 s.
```

```
| WORD SEARCH PUZZLE |
Masukkan nama file .txt yang diinginkan. (contoh ketikan: large1.txt)
Mohon pastikan tidak ada spasi di tempat yang tidak sesuai aturan masukan.
Masukkan nama file : small2.txt
```

output :

Masukan Puzzle :

L	V	P	B	A	F	U	C	E	Y	V	R	G	N	Z	N
R	U	A	D	X	Z	L	N	A	L	J	P	H	L	J	V
U	H	V	J	J	I	N	C	K	O	R	R	A	Z	J	Q
R	P	D	J	J	W	C	A	F	E	Q	E	H	H	M	U
N	T	M	J	W	K	E	F	L	O	O	D	I	E	Y	A
P	C	M	M	Y	D	T	A	A	I	P	E	A	X	P	N
X	W	G	H	I	D	T	A	A	G	D	E	U	U	Z	G
Z	D	M	N	X	D	T	A	Q	A	C	K	A	N	H	A
J	I	H	J	G	P	F	C	H	V	U	H	U	G	B	I
A	O	C	H	N	N	B	T	U	D	N	K	G	O	C	U
Y	A	C	H	A	A	T	A	C	P	Q	U	I	F	M	I
J	T	V	U	V	Y	A	T	U	M	A	L	X	F	M	U

- FURUHASHI
- SON
- WITAN
- CHANATHIP
- QUANGHAI
- KAMBUAYA
- VUVANTHAN
- BIHR
- YOSHIDA
- ASNAWI
- NADEO

[illegible]

> WORD : KAMBUAYA

[illegible]

```
>> WORD : SON
```

[illegible][illegible]

> WORD : QUANGHAI

[illegible]

> WORD : FURUHASHI

[illegible]

Jumlah huruf yang dicek : 851



[illegible][illegible]

```
>> Northeast direction
> WORD : TSUBASA
```

[illegible][illegible][illegible]

Matching time : 0.285303 s.

#### 4. Input medium1.txt

```
| WORD SEARCH PUZZLE |
```

Masukkan nama file .txt yang diinginkan. (contoh ketikan: large1.txt)  
Mohon pastikan tidak ada spasi di tempat yang tidak sesuai aturan masukan.

Masukkan nama file : medium1.txt

output :

Masukan Puzzle :

> WORD : -

A	N	J	J	S	O	N	U	L	Z	Z	N	J	J	U	K	T	N	Z	X
T	F	E	D	I	D	J	Z	I	K	U	D	C	Z	P	M	U	N	E	G
M	O	T	T	I	K	S	U	P	F	T	Z	W	C	E	L	G	X	D	C
M	I	L	K	I	N	S	I	D	E	Y	S	L	R	Y	C	C	C	Y	F
U	E	L	W	W	W	O	T	E	A	D	G	Y	J	F	W	Y	O	E	P
X	C	V	K	H	T	E	E	I	Q	N	F	I	C	V	C	C	T	N	Q
L	N	M	V	O	U	D	H	P	V	G	G	C	Z	U	L	B	T	R	L
C	G	X	E	P	U	Y	C	L	M	V	R	A	V	H	E	W	I	O	M
Q	P	S	L	G	M	T	A	B	Y	U	Q	A	N	M	G	G	U	T	X
P	D	R	A	W	B	Y	S	Q	M	E	L	J	Y	R	Q	V	F	T	H
C	Q	G	G	K	C	A	N	I	N	V	T	E	M	Q	O	O	Z	A	D
M	Q	R	F	F	W	V	Q	I	D	U	N	J	L	R	N	N	M	E	D
O	H	L	M	J	B	Q	H	F	P	E	O	G	Z	M	S	Y	P	C	E
V	T	Y	W	I	N	S	I	I	V	E	M	X	G	A	J	D	Z	A	Z
C	C	Y	M	I	A	S	D	Z	D	D	O	K	T	E	R	M	F	H	E
L	J	J	I	G	T	F	D	W	P	V	R	N	U	R	D	W	F	X	O
M	U	I	I	V	S	J	J	Y	D	M	I	K	M	S	L	A	D	W	X
K	I	M	I	G	T	S	H	I	N	E	P	O	B	R	A	D	I	N	N
N	I	E	H	L	O	H	Q	J	L	N	F	I	N	M	G	Q	K	G	V
K	A	R	M	E	H	V	E	L	I	V	C	L	T	S	U	J	B	K	P

Daftar kata yang dicari :

- MILKOUTSIDE
- MILKINSIDE
- DANGANRONPA
- ACEATTORNEY
- DGS
- OBRADINN
- OMORI
- KIMIGASHINE

[illegible]

```
>> Vertical Up direction
> WORD : ACEATTORNEY
```

```
>> Southeast direction
> WORD : MILKOUTSIDE
```

[illegible]

Jumlah huruf yang dicek : 1410

[illegible]

Jumlah huruf yang dicek : 582

```
5. Input medium2.txt
| WORD SEARCH PUZZLE |

Masukkan nama file .txt yang diinginkan. (contoh ketikan: large1.txt)
Mohon pastikan tidak ada spasi di tempat yang tidak sesuai aturan masukan.

Masukkan nama file : medium2.txt
```

```

Masukkan nama file : medium2.txt

Masukkan Puzzle :
> WORD : -
C P Y I Y K U L Z I Y V O I C R D U P A L Y C B S U K V U I J
U P Z Y D A L V D D M Z Q Q E D L E L E K F O O J I V F L J
M C B A D B V N W L P P O B J J Q J V L O H A X X C G U A F T N H B S O U A C C X I V X O A J R E D M B
I M I Q R M C Y H H D E S N N U Y P S T A L W L F Q Q Z Z S X V D R N I C I F O A F T N H B S O U A C C X I V X O A J R E D M B
S G Q Y I W B E M I I X V F N M V G K K H C F U W P X G N M D C G U L U D E G E P F U W Q Q A R I L Y C B S O U A C C X I V X O A J R E D M B
F Y K K O L K R I I X V F N M V G K K H C F U W P X G N M D C G U L U D E G E P F U W Q Q A R I L Y C B S O U A C C X I V X O A J R E D M B
V I A W R K R I I X V F N M V G K K H C F U W P X G N M D C G U L U D E G E P F U W Q Q A R I L Y C B S O U A C C X I V X O A J R E D M B
B U A W R K R I I X V F N M V G K K H C F U W P X G N M D C G U L U D E G E P F U W Q Q A R I L Y C B S O U A C C X I V X O A J R E D M B
J F L L U C Y I F N M V G K K H C F U W P X G N M D C G U L U D E G E P F U W Q Q A R I L Y C B S O U A C C X I V X O A J R E D M B
C Q C F Y T M U L X U Z K F P U A L A R I L Y C B S O U A C C X I V X O A J R E D M B
I C U Y T M U L X U Z K F P U A L A R I L Y C B S O U A C C X I V X O A J R E D M B
M B U L X U Z K F P U A L A R I L Y C B S O U A C C X I V X O A J R E D M B
J B U L X U Z K F P U A L A R I L Y C B S O U A C C X I V X O A J R E D M B
H V V Y U Z K F P U A L A R I L Y C B S O U A C C X I V X O A J R E D M B

```

```
HASIL PENCARIAN :
>> Horizontal Right direction
> WORD : VIOLIN
```





```
>> Northwest direction  
> WORD : BASS
```

-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	S	-	-	-	-
-	-	-	-	-	-	-	-	S	-	-	-	-
-	-	-	-	-	-	-	-	-	A	-	-	-
-	-	-	-	-	-	-	-	-	-	B	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-

```
Jumlah huruf yang dicek : 204  
Matching time : 0.5675868 s.
```

6. Input medium3.txt

```
| WORD SEARCH PUZZLE |
Masukkan nama file .txt yang diinginkan. (contoh ketikan: large1.txt)
Mohon pastikan tidak ada spasi di tempat yang tidak sesuai aturan masukan.
Masukkan nama file : medium3.txt
```

output :

```
> WORD : -
```

C	J	Z	A	I	R	E	T	A	L	E	T	A	I	C	E	V	O	J	D
T	U	A	O	D	A	O	D	Z	S	N	P	Z	S	I	O	S	L	O	A
R	L	T	I	G	I	W	B	D	E	X	V	D	E	Y	F	G	Q	V	V
R	Y	Z	Y	U	U	H	W	E	L	T	P	E	L	H	X	K	G	B	I
W	S	J	H	O	M	F	C	T	R	E	C	L	R	M	X	Z	L	I	E
M	S	M	K	E	Q	J	V	U	C	F	Y	I	W	J	F	N	Q	T	B
S	Z	Z	G	W	D	O	B	B	O	T	O	O	R	X	Z	D	G	R	G
P	Q	F	O	K	T	D	Y	V	H	Q	A	A	F	J	O	V	X	F	X
T	D	C	V	H	T	B	Y	J	F	J	O	H	I	O	I	P	P	V	X
L	Q	Q	V	N	Y	G	Q	Z	U	I	C	L	I	R	X	M	Y	I	B
M	L	H	N	P	C	I	Q	Z	E	V	Y	G	C	B	D	O	X	S	R
M	L	J	Q	C	S	I	Q	Z	I	I	P	A	I	N	D	J	N	V	R
X	P	J	Q	F	U	J	T	I	J	O	V	K	W	N	P	J	K	X	W
D	J	Q	D	E	W	K	T	C	K	M	V	I	G	M	M	M	V	A	R
Z	K	G	P	A	N	K	S	B	P	Y	O	G	I	L	M	M	E	N	I
C	A	T	R	M	R	P	E	D	I	I	G	S	T	X	C	F	A	C	G
T	H	S	O	S	C	B	R	U	K	B	I	X	C	K	U	N	X	B	Y
S	C	I	J	B	A	I	Z	P	I	H	I	C	E	Q	T	A	X	G	F

- LAHM
- TRENT
- UCHIDA
- DANIALVES
- AZPILICUETA
- DAVIES
- ROBERTOCARLOS
- HAKIMI

```
>> Horizontal Right direction
```

L A H M

```
>> Horizontal Left direction
```

S E V L A I N A D

[illegible]

Jumlah huruf yang dicek : 1974

[illegible]

Djumlah huruf yang dicek : 1427

[illegible]

Jumlah huruf yang dicek : 796

```
T  
T  
R  
E  
N
```

Jumlah huruf yang dicek : 428

```
>> Southwest direction
Jumlah huruf yang dicek : 418
```

\_\_\_\_\_

[illegible]

Jumlah huruf yang dicek : 222

Matching time : 0.5354113 s.

```
| WORD SEARCH PUZZLE |

Masukkan nama file .txt yang diinginkan. (contoh ketikan: large1.txt)
Mohon pastikan tidak ada spasi di tempat yang tidak sesuai aturan masukan.

Masukkan nama file : large1.txt
```

```
Masukkan nama file .txt yang diinginkan. (contoh ketikan: large1.txt)
Mohon pastikan tidak ada spasi di tempat yang tidak sesuai aturan masukan.
Masukkan nama file : large1.txt
```

```
Mohon pastikan tidak ada spasi di tempat yang tidak sesuai aturan masukan.
Masukkan nama file : large1.txt
```

Masukkan nama file : large1.txt

output :

Hasukan Puzzle :	
> WORD :	-
S	R
C	R
Y	N
U	T
T	O
C	F
T	P
Q	D
P	H
F	V
Z	L
M	H
B	M
V	E
G	H
M	U
C	M
O	O
I	O
I	Y
I	Z
F	T
T	X
U	L
I	X
E	I
F	N
K	B
F	L
R	C
Q	J
G	W
H	P
P	N
E	O
S	K
S	Q
W	T
N	Y
F	G
D	Q
K	V
P	I
O	K
B	T
T	P
I	A
A	S
S	T
T	J
J	D
Q	Q
Q	D
U	P
F	N
N	M
M	U
U	B
B	Y
O	N
A	X
K	O
Q	S
R	H
Z	N
O	Q
F	N
S	E
H	E
V	P
I	P
D	S
S	Y
A	L
O	E
I	O
I	Y
I	Z
F	T
T	X
U	L
I	X
E	I
F	N
K	B
F	L
R	C
Q	J
G	W
H	P
P	N
E	O
S	K
S	Q
W	T
N	Y
F	G
D	Q
K	V
P	I
O	K
B	T
T	P
I	A
A	S
S	T
T	J
J	D
Q	Q
Q	D
U	P
F	N
N	M
M	U
U	B
B	Y
O	N
A	X
K	O
Q	S
R	H
Z	N
O	Q
F	N
S	E
H	E
V	P
I	P
D	S
S	Y
A	L
O	E
I	O
I	Y
I	Z
F	T
T	X
U	L
I	X
E	I
F	N
K	B
F	L
R	C
Q	J
G	W
H	P
P	N
E	O
S	K
S	Q
W	T
N	Y
F	G
D	Q
K	V
P	I
O	K
B	T
T	P
I	A
A	S
S	T
T	J
J	D
Q	Q
Q	D
U	P
F	N
N	M
M	U
U	B
B	Y
O	N
A	X
K	O
Q	S
R	H
Z	N
O	Q
F	N
S	E
H	E
V	P
I	P
D	S
S	Y
A	L
O	E
I	O
I	Y
I	Z
F	T
T	X
U	L
I	X
E	I
F	N
K	B
F	L
R	C
Q	J
G	W
H	P
P	N
E	O
S	K
S	Q
W	T
N	Y
F	G
D	Q
K	V
P	I
O	K
B	T
T	P
I	A
A	S
S	T
T	J
J	D
Q	Q
Q	D
U	P
F	N
N	M
M	U
U	B
B	Y
O	N
A	X
K	O
Q	S
R	H
Z	N
O	Q
F	N
S	E
H	E
V	P
I	P
D	S
S	Y
A	L
O	E
I	O
I	Y
I	Z
F	T
T	X
U	L
I	X
E	I
F	N
K	B
F	L
R	C
Q	J
G	W
H	P
P	N
E	O
S	K

```
Daftar kata yang dicari :
- CHAINSAWMAN
- YOUZITSU
- GIANTKILLING
- YONA
- TOWEROFGOD
- GRANDBLUE
- JOJO
-----
```

```

HASIL PENGACARAN :
-> Horizontal Right direction
> WORD : YOUZITSU

Y O U Z I T S U

Jumlah huruf yang dicek : 6164

```

```
>> Horizontal Left direction
Jumlah huruf yang dicek : 4373
>> Vertical Down direction
> WORD : YONA
Jumlah huruf yang dicek : 2738

>> Vertical Up direction
> WORD : GIANTKILLING
Jumlah huruf yang dicek : 2331

>> Northwest direction
Jumlah huruf yang dicek : 631
Matching time : 0.8172704 s.
```

8. Input large2.txt

```
| WORD SEARCH PUZZLE |

Masukkan nama file .txt yang diinginkan. (contoh ketikan: large1.txt)
Mohon pastikan tidak ada spasi di tempat yang tidak sesuai aturan masukan.

Masukkan nama file : large2.txt
```

output :

> WORD : -

[illegible]

- ZHUGELIANG

- LUXUN

- ZHOU YU  
- HUANG- LUBU  
CLUNG

- LIUBEI

```
> WORD : LUXUN
| - - - - -
```

1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30

[illegible]

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479	480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520	521	522	523	524
--	---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30

\_\_\_\_\_

	1	2	3	4
1				
2				
3				
4				


	1	2	3	4
1				
2				
3				
4				

	1	2	3	4
1				
2				
3				

	1	2	3	4
1				
2				
3				

	1	2	3	4
1	1	2	3	4
2	2	1	3	4
3	3	3	1	4
4	4	4	4	1



\_\_\_\_\_

```
>> Horizontal left direction
> WORD : HURANGAI
[Grid with word HURANGAI highlighted horizontally]
Jumlah huruf yang dicek : 5451

>> Vertical Down direction
> WORD : ZHOOUU
[Grid with word ZHOOUU highlighted vertically]
Jumlah huruf yang dicek : 4270

>> vertical up direction
Jumlah huruf yang dicek : 3668
>> Southeast direction
> WORD : LIUBEI
[Grid with word LIUBEI highlighted southeast]
Jumlah huruf yang dicek : 2968

>> Northeast direction
> WORD : LUBU
[Grid with word LUBU highlighted northeast]
Jumlah huruf yang dicek : 1840

>> Southwest direction
> WORD : CASPI
[Grid with word CASPI highlighted southwest]
Jumlah huruf yang dicek : 883

>> Northwest direction
> WORD : ZHUSILANG
[Grid with word ZHUSILANG highlighted northwest]
Jumlah huruf yang dicek : 285
Matching time : 0.9603147 s.
```

9. Input large3.txt

```
| WORD SEARCH PUZZLE |

Masukkan nama file .txt yang diinginkan. (contoh ketikan: large1.txt)
Mohon pastikan tidak ada spasi di tempat yang tidak sesuai aturan masukan.

Masukkan nama file : large3.txt
```

output :





S  
E  
N  
T  
O  
U  
C  
A  
M  
P  
R  
H  
I  
T  
A  
B  
L

```

to Southeast direction
W80D - AYANAKOUIKYOTAKA
A Y A N O K U T I K I Y O T A K A

```

```
>> Northeast direction
Jumlah huruf yang dicek : 1202
```

OKUOTKAZO A

```
>> Southwest direction
WORD : ANARAGIKOYORI
A
R
A
R
G
I
K
O
Y
O
I
H
I
```

Jumlah huruf yang dicek : 662

